```python
# -*- coding: utf-8 -*-
"""
Created on Tue Mar  5 10:48:44 2019

@author: Nate
"""

import numpy as np
import matplotlib.pyplot as plt
import pdb

#number of trials
N = 10000


#α=0.05 to 0.5 in increments of 0.05
alpha = [i*.05 for i in range(1,11)]
sigma = [4*i for i in alpha]
energy = []
std_dev = []

##################################################################
##################################################################
# NORMAL SAMPLES USING BOX-MUELLER METHOD
# DRAW SAMPLES FROM PROPOSAL DISTRIBUTION

for j in range(len(alpha)):
    u = np.array([])
    w = np.array([])
    u2 = np.array([])
    w2 = np.array([])

    for i in range(N):
        u = np.append(u ,np.random.uniform(0,1))
        w = np.append(w ,np.random.uniform(0,1))
        u2 = np.append(u2 ,np.random.uniform(0,1))
        w2 = np.append(w2 ,np.random.uniform(0,1))

    r = np.array([np.sqrt(-np.log(i)/(2*alpha[j])) for i in u])
    theta = np.array([2*np.pi*i for i in w])
    r2 = np.array([np.sqrt(-np.log(i)/(2*alpha[j])) for i in u2])
    theta2 = np.array([2*np.pi*i for i in w2])

    x = np.array([])
    y = np.array([])
    z = np.array([])

    for i in range(N):
        x = np.append(x, np.array(r[i]*np.cos(theta[i])))
        y = np.append(y, np.array(r[i]*np.sin(theta[i])))
        z = np.append(z, np.array(r2[i]*np.cos(theta2[i])))


##################################################################
##################################################################
#Calculating Energy

    en = 0

    for i in range(N):
        r_mag = np.linalg.norm([x[i], y[i], z[i]])
        en += alpha[j]*(3-2*alpha[j]*r_mag**2)-1/r_mag

    energy.append(en/N)

    en_avg = en/N

```

```
    to_sum = 0

    for i in range(N):
        r_mag = np.linalg.norm([x[i], y[i], z[i]])
        to_sum += (alpha[j]*(3-2*alpha[j]*r_mag**2)-1/r_mag)**2

    std_dev.append(np.sqrt(to_sum)/N)

print(std_dev)
################################################################
################################################################
#Plotting


fig1, axes = plt.subplots(2, 3)
fig1.subplots_adjust(top=0.92, left=0.07, right=0.97, hspace=0.3, wspace=0.3)
((ax1, ax2, ax3), (ax4, ax5, ax6)) = axes  # unpack the axes

ax1.hist(theta,100)
ax1.set_ylabel('Counts')
ax1.set_xlabel('$theta_i$')
ax1.set_title("Theta", va='bottom')

ax2.hist(r,100)
ax2.set_ylabel('Counts')
ax2.set_xlabel('$R_i$')
ax2.set_title("R", va='bottom')

ax3.hist(r2,100)
ax3.set_ylabel('Counts')
ax3.set_xlabel('$R_i$')
ax3.set_title("R2", va='bottom')

ax4.hist(x,100)
ax4.set_ylabel('Counts')
ax4.set_xlabel('$x_i$')
ax4.set_title("X", va='bottom')

ax5.hist(y,100)
ax5.set_ylabel('Counts')
ax5.set_xlabel('$y_i$')
ax5.set_title("Y", va='bottom')

ax6.hist(z,100)
ax6.set_ylabel('Counts')
ax6.set_xlabel('$z_i$')
ax6.set_title("Z", va='bottom')



fig2, axes2 = plt.subplots()


print('alpha: ', alpha, '    |    Energy is: ', [3/2*num-(2**(3/2))/(np.pi**.5)*num**.5
for num in alpha])


#axes2.scatter(alpha, [3/2*num-(2**(3/2))/(np.pi**.5)*num**.5 for num in alpha])
axes2.errorbar(alpha, [3/2*num-(2**(3/2))/(np.pi**.5)*num**.5 for num in alpha],
yerr=std_dev, fmt='o')
axes2.scatter(8/(9*np.pi), 3/2*(8/(9*np.pi))-2**(3/2)/(np.pi**.5)*(8/(9*np.pi))**.5,
color='red')

axes2.plot(alpha, energy)
axes2.set_ylabel('Energy')
axes2.set_xlabel('alpha')
axes2.set_title("Energy vs alpha", va='bottom')
```

```
132
133
134   plt.show()
```